Minhaj University Lahore

Assignment No : 2		Mid ○	Final ●
Course Title & Code	Computer Organization and Assembly Language		
Submitted By:	Atif Nasir		
Registration No:	2023f-mulbscs-059		
Semester/Class/ Section:	4th || Bscs || "B"		
Submitted To:	Sir . Muhammad Irfan Akram		
Due Date	Online	Hard copy	
	yes	yes	
Student's Signature:			

Note: *Please avoid cutting/ overwriting in any of the above fields.
*Complete the task on standard A4 size papers/assignment pages.

***For Instructor's use only.**

Total Marks:	
Obtained Marks:	
Signatures:	

Faculty of CS & IT,Minhaj University Lahore

Assignment Detail:

1. **Scenario #1:** *A robot controller is programmed to pick two sensor values and store them temporarily before comparison.*

Q: Which assembly instructions would you use to transfer the values from the sensors to temporary storage (registers or memory)?Explain your choice.

2. **Scenario #2** :*An encryption algorithm rotates bits of a byte for obfuscation. You are asked to rotate the bits of a value to the left by 2 positions.*

Q: Which instruction(s) would you use and why? What's the effect of rotating instead of shifting?

Scenario #1: Sensor Value Transfer to Temporary Storage

Q: *Which assembly instructions would you use to transfer the values from the sensors to temporary storage (registers or memory)? Explain your choice. (5 Marks)*

Answer:

To transfer values from two sensors into temporary storage, the best choice in assembly language is the MOV instruction. It allows moving data from memory-mapped I/O (sensors) into CPU registers or memory variables for further processing.

Example Program: Transfer Two Sensor Values to Registers

Program Code:

```
.MODEL SMALL
.STACK 100H
.DATA
Sensor1 DB 5AH    ; Simulated Sensor 1 value
Sensor2 DB 3FH    ; Simulated Sensor 2 value
Temp1   DB ?      ; Temporary storage 1
Temp2   DB ?      ; Temporary storage 2

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX

   MOV AL, Sensor1   ; Move Sensor1 value to AL register
   MOV Temp1, AL     ; Store AL in Temp1

   MOV BL, Sensor2   ; Move Sensor2 value to BL register
   MOV Temp2, BL     ; Store BL in Temp2
```

```
    ; Comparison can be done later if needed
    ; CMP Temp1, Temp2

    MOV AH, 4CH
    INT 21H
MAIN ENDP
END MAIN
```

Why use MOV?

- Fast and efficient: Directly accesses memory or registers.
- Non-destructive: Doesn't alter the value being moved.
- Perfect for temporary storage: Allows values to be held during processing or comparison.

Scenario #2: Bitwise Left Rotation for Encryption

Q: Which instruction(s) would you use and why? What's the effect of rotating instead of shifting? (5 Marks)

Answer:

To rotate a byte's bits to the left by 2 positions, the ROL instruction (Rotate Left) is used. This operation rotates all bits within a byte and wraps the leftmost bits back to the rightmost positions.

Example Program: Rotate Byte to Left by 2 Bits

Program Code:

```
.MODEL SMALL
.STACK 100H
.DATA
Original DB 96H     ; Binary: 1001 0110
Result   DB ?       ; Store result after rotation

.CODE
MAIN PROC
    MOV AX, @DATA
    MOV DS, AX

    MOV AL, Original ; Load value into AL
    ROL AL, 2        ; Rotate AL left by 2 bits
    MOV Result, AL   ; Store result
```

```
    MOV AH, 4CH
    INT 21H
MAIN ENDP
END MAIN
```

ROL vs SHL: What's the Difference?

Effect	Operation
SHL (Shift Left)	Pushes bits to the left, fills with zeros on the right. Loses overflow bits.
ROL (Rotate Left)	Pushes bits left, but cycles the overflow bits back to the right. No data loss.

Effect of ROL AL, 2 **on 96H (1001 0110):**
After rotation: A5H (1010 0101)
The leftmost two bits 10 are wrapped around to the right.